J 13

```
   SSSSSSSSSSSS   YYY         YYY     SSSSSSSSSSSS     IIIIIIIII     NNN         NNN     IIIIIIIII
   SSSSSSSSSSSS   YYY         YYY     SSSSSSSSSSSS     IIIIIIIII     NNN         NNN     IIIIIIIII
   SSSSSSSSSSSS   YYY         YYY     SSSSSSSSSSSS     IIIIIIIII     NNN         NNN     IIIIIIIII
SSS               YYY         YYY  SSS                    III        NNN         NNN        III
SSS               YYY         YYY  SSS                    III        NNN         NNN        III
SSS               YYY         YYY  SSS                    III        NNN         NNN        III
SSS                  YYY   YYY     SSS                    III        NNNNNN      NNN        III
SSS                  YYY   YYY     SSS                    III        NNNNNN      NNN        III
SSS                  YYY   YYY     SSS                    III        NNNNNN      NNN        III
   SSSSSSSSS            YYY           SSSSSSSSS           III        NNN   NNN   NNN        III
   SSSSSSSSS            YYY           SSSSSSSSS           III        NNN   NNN   NNN        III
   SSSSSSSSS            YYY           SSSSSSSSS           III        NNN   NNN   NNN        III
            SSS         YYY                    SSS        III        NNN      NNNNNN        III
            SSS         YYY                    SSS        III        NNN      NNNNNN        III
            SSS         YYY                    SSS        III        NNN      NNNNNN        III
            SSS         YYY                    SSS        III        NNN         NNN        III
            SSS         YYY                    SSS        III        NNN         NNN        III
            SSS         YYY                    SSS        III        NNN         NNN        III
SSSSSSSSSSSS            YYY        SSSSSSSSSSSS        IIIIIIIII     NNN         NNN     IIIIIIIII
SSSSSSSSSSSS            YYY        SSSSSSSSSSSS        IIIIIIIII     NNN         NNN     IIIIIIIII
SSSSSSSSSSSS            YYY        SSSSSSSSSSSS        IIIIIIIII     NNN         NNN     IIIIIIIII
```

```
**FILE**ID**SYSMOU                                   G  4
```

```
  SSSSSSSS  YY      YY    SSSSSSSS  MM      MM    OOOOOO    UU      UU
  SSSSSSSS  YY      YY    SSSSSSSS  MM      MM    OOOOOO    UU      UU
SS          YY      YY  SS          MMMM  MMMM  OO      OO  UU      UU
SS          YY      YY  SS          MMMM  MMMM  OO      OO  UU      UU
SS            YY  YY    SS          MM  MM  MM  OO      OO  UU      UU
SS            YY  YY    SS          MM  MM  MM  OO      OO  UU      UU
  SSSSSS        YY        SSSSSS    MM      MM  OO      OO  UU      UU
  SSSSSS        YY        SSSSSS    MM      MM  OO      OO  UU      UU
        SS      YY              SS  MM      MM  OO      OO  UU      UU
        SS      YY              SS  MM      MM  OO      OO  UU      UU
        SS      YY              SS  MM      MM  OO      OO  UU      UU      ....
        SS      YY              SS  MM      MM  OO      OO  UU      UU      ....
SSSSSSSS        YY      SSSSSSSS    MM      MM    OOOOOO    UUUUUUUUUU      ....
SSSSSSSS        YY      SSSSSSSS    MM      MM    OOOOOO    UUUUUUUUUU      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

SYS
VO

```
0 MODULE SYSMOU (
0                LANGUAGE (BLISS32),
0                IDENT = 'V04-000'
0                ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  MOUNT Utility Structure Levels 1 & 2
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains the code and data needed to mount the system
1 !       disk during system initialization.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  1-Nov-1977  19:02
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-010 CDS0005         Christian D. Saether    29-Aug-1984
1 !               Use STAND_ALONE_REBUILD routine to avoid unnecessary
1 !               rebuilds.
1 !
1 !       V03-009 CDS0004         Christian D. Saether     2-Aug-1984
1 !               Test the sysgen flag REBLDSYSD to determine whether
```

```
1 !                 rebuild should be performed.
1 !
1 !         V03-008 HH0041          Hai Huang               24-Jul-1984
1 !                 Remove REQUIRE 'OBJD$:[VMSLIB.OBJ]MOUNTMSG.REQ'.
1 !
1 !         V03-007 HH0018          Hai Huang               06-May-1984
1 !                 Use $GETDVI to obtain the physical device name of the
1 !                 system device.
1 !
1 !         V03-006 TMH0006         Tim Halvorsen           14-Apr-1984
1 !                 Add MOUNT_FLAGS to list of dummy storage needed for
1 !                 linked-in-MOUNT.
1 !
1 !         V03-005 CDS0003         Christian D. Saether    19-Oct-1983
1 !                 Now that volume rebuild works, allow FID and EXT caching.
1 !
1 !         V03-004 TCM0001         Trudy C. Matthews       19-Aug-1983
1 !                 Interlock mounts of the system disk with other potential
1 !                 mounters of the same disk in the cluster.  Add cluster
1 !                 consistency checking routines.
1 !
1 !         V03-003 CDS0002         Christian D. Saether    15-Aug-1983
1 !                 Set OPT_NOEXT_C, OPT_NOFID_C, OPT_NOQUO_C, and OPT_WTHRU
1 !                 to REALLY disable caching.
1 !
1 !         V03-002 CDS0001         Christian D. Saether     5-Aug-1983
1 !                 Temporarily disable caching on system disk until
1 !                 xqp cluster rebuild works.
1 !
1 !         V03-001 STJ3061         Steven T. Jeffreys,     04-Mar-1983
1 !                 Added definitions of DEVICE_INDEX and CALLERS_ACMOD.
1 !                 These parallel definitions in VMOUNT.
1 !
1 !         V02-008 STJ0202         Steven T. Jeffreys,     05-Feb-1982
1 !                 Make sure the OPT_MOUNTVER bit gets set.  The first
1 !                 attempt at this ended in disaster.
1 !
1 !         V02-007 STJ0175         Steven T. Jeffreys,     06-Jan-1982
1 !                 Set up the database to ensure the system disk
1 !                 is a candidate for mount verification.
1 !
1 !         V02-006 ACG0248         Andrew C. Goldstein,    31-Dec-1981  16:56
1 !                 Use default logical name, fix use of $GETDEV
1 !
1 !         V02-005 ACG0181         Andrew C. Goldstein,    13-Oct-1980  15:37
1 !                 Fix cross facility references
1 !
1 !         V0104   ACG0123         Andrew C. Goldstein,    12-Feb-1980  18:23
1 !                 Integrate disk rebuild into MOUNT
1 !
1 !         V0103   ACG0079         Andrew C. Goldstein,    11-Nov-1979  19:32
1 !                 MOUNT changes for write-back cacheing
1 !
1 !         V0102   ACG0072         Andrew C. Goldstein,    22-Oct-1979  13:53
1 !                 Check primary and secondary device char
1 !
1 !         V101    ACG0003         Andrew C. Goldstein,  28-Dec-1978  15:23
```

```
;  115        0115  1 !          Add global variables for multi-volume MOUNT
;  116        0116  1 !
;  117        0117  1 !          V100    ACG0001          Andrew C. Goldstein,  28-Dec-1978  15:22
;  118        0118  1 !          Previous revision history moved to SYSINIT.REV
;  119        0119  1 !**
;  120        0120  1
;  121        0121  1
;  122        0122  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;  123        0123  1 REQUIRE 'LIB$:MOUDEF.B32';
;  124        0655  1
;  125        0656  1
;  126        0657  1 FORWARD ROUTINE
;  127        0658  1          MOUNT_SYSTEM,                       ! main routine
;  128        0659  1          MAIN_HANDLER;                       ! condition handler
```

```
!+
!
! Own storage for general use in the MOUNT utility
!
!-

GLOBAL
        STORED_CONTEXT   : BITVECTOR [32],! store the context of some 1 time only
        MOUNT_FLAGS      : LONG INITIAL(0), ! MOUNT flags
        LOCK_STATUS      : VECTOR [2],    ! Lock status block for $ENQ call
        DEVICE_INDEX     : LONG,          ! index into PHYS_NAME array
        CALLERS_ACMOD    : LONG,          ! caller's access mode
        CLEANUP_FLAGS    : BITVECTOR [32], ! error cleanup status flags
        CHANNEL,                          ! channel number for I/O
        MAILBOX_CHANNEL,                  ! channel number of ACP termination mailbox
        PHYS_BUFFER      : VECTOR [20, BYTE],
                                          ! buffer to construct phys device name
        PHYS_NAME        : VECTOR [2]
                           INITIAL (0, PHYS_BUFFER), ! descriptor of physical device name
        LOG_BUFFER       : VECTOR [20, BYTE],
                                          ! buffer to construct logical name
        HOME_BLOCK       : BBLOCK [512], ! buffer for volume header label or home block
        DEVICE_CHAR      : BBLOCK [DIB$K_LENGTH],
                                          ! buffer for device characteristics
        DEVICE_CHAR2     : BBLOCK [DIB$K_LENGTH],
                                          ! buffer for 2nd device characteristics
        HOMEBLOCK_LBN,                    ! LBN of home block read
        HEADER_LBN,                       ! LBN of file header
        DEV_INDEX,                        ! index into device data table
        USER_STATUS      : VECTOR [2],    ! status return for various routines
        CURRENT_RVN,                      ! RVN of volume being mounted
        CURRENT_VCB      : REF BBLOCK,    ! address of VCB used by CHECK_HEADER2
        REAL_RVT         : REF BBLOCK,    ! address of RVT allocated for volume set
        REAL_VCB         : REF BBLOCK,    ! address of VCB allocated for volume
        REAL_FCB         : REF BBLOCK,    ! address of FCB allocated for volume
        REAL_WCB         : REF BBLOCK,    ! address of window allocated for volume
        REAL_VCA         : REF BBLOCK,    ! address of cache block allocated
        REAL_AQB         : REF BBLOCK,    ! address of AQB allocated for volume
        LOG_ENTRY        : REF BBLOCK,    ! address of logical name entry
        MTL_ENTRY        : REF BBLOCK,    ! address of mounted volume list entry
        SLOG_ENTRY       : REF BBLOCK,    ! address of volume set logical name entry
        SMTL_ENTRY       : REF BBLOCK,    ! address of volume set mounted volume list entry

        DEVCHAR_DESC     : VECTOR [2] INITIAL (DIB$K_LENGTH, DEVICE_CHAR),
                                          ! descriptor for device characteristics
        DEVCHAR_DESC2    : VECTOR [2] INITIAL (DIB$K_LENGTH, DEVICE_CHAR2);
                                          ! descriptor for device characteristics



!+
!
! The following area is a hand crafted mount parser output suitable for
! mounting the system disk.
!
!-

```

```
1 GLOBAL
1         MOUNT_OPTIONS     : BITVECTOR [64] ! option flags
1                           INITIAL (
2                                   (1^OPT_SYSTEM OR          ! First 32 bits
2                                    1^OPT_WRITE OR
2                                    1^OPT_BLOCK OR
2                                    1^OPT_OVR_ID OR
2                                    1^OPT_DEVICE OR
1                                    1^OPT_LABEL),
2                                   (1^(OPT_MOUNTVER-32) OR          ! Last 32 bits
2                                     1^(OPT_NOQUO_C-32) OR
2                                     1^(OPT_WTHRU-32))
1                                   ),
1
1         PROTECTION        : INITIAL (0),  ! value of /PROTECTION switch
1         OWNER_UIC         : INITIAL (0),  ! value of /OWNER_UIC switch
1         USER_UIC          : INITIAL (0),  ! value of /USER_UIC switch
1         EXTENSION         : INITIAL (0),  ! value of /EXTENSION switch
1         WINDOW            : INITIAL (0),  ! value of /WINDOW switch
1         ACCESSED          : INITIAL (0),  ! value of /ACCESSED switch
1         BLOCKSIZE         : INITIAL (0),  ! value of /BLOCK switch
1         EXT_CACHE         : INITIAL (0),  ! value of /CACHE=(EXTENT=n) switch
1         FID_CACHE         : INITIAL (0),  ! value of /CACHE=(FILE=n) switch
1         QUO_CACHE         : INITIAL (0),  ! value of /CACHE=(QUOTA=n) switch
1         EXT_LIMIT         : INITIAL (0),  ! value of /CACHE=(LIMIT=n) switch
1         DEVICE_COUNT      : INITIAL (1),  ! number of devices specified
1         LABEL_COUNT       : INITIAL (1),  ! number of volume labels specified
1         LOG_NAME          : VECTOR [2],   ! logical name of system disk
1         STRUCT_NAME       : VECTOR [2],   ! descriptor of structure name
1         VID_STRING        : VECTOR [2],   ! descriptor of VISUAL_ID string
1         COMMENT_STRING    : VECTOR [2],   ! descriptor of COMMENT string
1         ACP_STRING        : VECTOR [2],   ! descriptor of ACP device or name string
1         DRIVE_COUNT       : VECTOR [1],   ! value of /DRIVES switch
1
1         PARSE_IMP_END     : VECTOR [0];   ! end of data area
1
1 GLOBAL BIND
1         LABEL_STRING      = DESCRIPTOR ('SYSTEMDISK') : VECTOR;
1                                           ! dummy volume label of system disk
```

```
1 GLOBAL ROUTINE MOUNT_SYSTEM (SYS_CHANNEL) =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine mounts the system disk (i.e., the disk to which the
1 !       channel is assigned) and starts the ACP.
1 !
1 !
1 ! CALLING SEQUENCE:
1 !       MOUNT_SYSTEM (ARG1)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: channel number assigned to disk
1 !
1 ! IMPLICIT INPUTS:
1 !       own storage of this module
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       1 if successful, assorted statuses if not
1 !
1 ! SIDE EFFECTS:
1 !       system disk mounted, ACP started, logical name created
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2       MOUNT_IOSB      : VECTOR [2],
2       ALLDEVNAM_BUF   : VECTOR [NAMEBUF_LEN, BYTE]
2                         INITIAL (BYTE ('MOU$', REP NAMEBUF_LEN-4 OF (' '))),
2       ALLDEVNAM_DESC  : VECTOR [2] INITIAL (0, ALLDEVNAM_BUF),
2       DEVICE_ITMLST   : BBLOCK [(2 * 12) + 4] INITIAL
2
2                         !
2                         ! 1st item - device name
2                         !
2                         (WORD (20),             ! Length of dev name buffer
2                          WORD (DVI$_DEVNAM),    ! Item code for device name
2                          LONG (PHYS_BUFFER),    ! Dev name buffer address
2                          LONG (PHYS_NAME),      ! Returned dev name length
2                         !
2                         ! 2nd item - allocation class name
2                         !
2                          WORD (NAMEBUF_LEN - 4),
2                          WORD (DVI$_ALLDEVNAM),
2                          LONG (ALLDEVNAM_BUF + 4),
2                          LONG (ALLDEVNAM_DESC),
2                         !
```

```
                                        ! End of list.
                                        !
                                          LONG (0)),
                STATUS,                                 ! system service status
                P;                                      ! pointer into characteristics block

        EXTERNAL
                EXE$GL_STATIC_FLAGS : ADDRESSING_MODE (GENERAL) BITVECTOR [32],
                DEV_CTX             : BBLOCK FIELD (DC);
                                                        ! device value block context fields

        EXTERNAL LITERAL
                EXE$V_REBLDSYSD;

        EXTERNAL ROUTINE
                READ_HOMEBLOCK,                         ! read disk home block
                MOUNT_DISK1,                            ! mount disk, level 1
                MOUNT_DISK2,                            ! mount disk, level 2
                STAND_ALONE_REBUILD,                    ! rebuild disk bitmaps and quota file
                GET_DEVICE_CONTEXT;                     ! get device lock value block context

        ! Enable the condition handler.
        !

        ENABLE MAIN_HANDLER;

        CALLERS_ACMOD = PSL$C_SUPER;                    ! used for logical name access mode
        CHANNEL = .SYS_CHANNEL;

        !
        ! Take out a lock to synchronize all mounts of this device in a cluster.
        ! First we must construct the lock resource name (use the allocation class
        ! name returned by $GETDVI).
        !

        STATUS = $GETDVIW (CHAN   = .CHANNEL,
                           ITMLST = DEVICE_ITMLST,
                           EFN    = MOUNT_EFN,
                           IOSB   = MOUNT_IOSB);
        IF NOT .STATUS THEN ERR_EXIT (.STATUS);
        ALLDEVNAM_DESC[0] = .ALLDEVNAM_DESC[0] + 4;

        STATUS = $ENQW (LKMODE = LCK$K_EXMODE,
                        LKSB   = LOCK_STATUS,
                        FLAGS  = LCK$M_SYSTEM,
                        RESNAM = ALLDEVNAM_DESC,
                        EFN    = MOUNT_EFN,
                        ACMODE = PSL$C_EXEC);
        IF NOT .STATUS THEN ERR_EXIT (.STATUS);
        !
        ! Get the device characteristics and do device type validation: Make sure
        ! the device is mountable at all, and check that the mount qualifiers are
        ! consistent with the device type. A mismatch between primary and secondary
        ! device characteristics indicates a spooled device or something else strange.
        ! Reject such.
        !
```

```
2
2 $GETCHN (CHAN = .CHANNEL, PRIBUF = DEVCHAR_DESC, SCDBUF = DEVCHAR_DESC2);
2
2 IF CH$NEQ (DIB$K_LENGTH, DEVICE_CHAR, DIB$K_LENGTH, DEVICE_CHAR2, 0)
2 OR NOT .DEVICE_CHAR[DEV$V_FOD]
2 THEN ERR_EXIT (SS$_NOTFILEDEV);
2
2 IF NOT .DEVICE_CHAR[DEV$V_AVL]
2 THEN ERR_EXIT (SS$_DEVOFFLINE);
2
2 IF .DEVICE_CHAR[DEV$V_MNT]
2 THEN ERR_EXIT (SS$_DEVMOUNT);
2
2 IF .DEVICE_CHAR[DEV$V_SQD]
2 THEN ERR_EXIT (SS$_NOTFILEDEV);
2
2 !
2 ! The following is for reference only. The physical device name is now
2 ! obtained with the $GETDVIW system service, rather than formatting device
2 ! name and the unit number.
2 !
2 ! Construct the physical device name by appending the ascii unit number to
2 ! the device name in the device characteristics.
2 !
2 !
2 !PHYS_NAME[0] = 20;
2 !PHYS_NAME[1] = PHYS_BUFFER;
2 !$FAO (
2 !       DESCRIPTOR ('_!AC!UW:'),
2 !       PHYS_NAME[0],
2 !       PHYS_NAME[0],
2 !       DEVICE_CHAR + .DEVICE_CHAR[DIB$W_DEVNAMOFF],
2 !       .DEVICE_CHAR[DIB$W_UNIT]
2 !       );
2
2 ! Now attempt to read the home block or volume header label, as appropriate
2 ! for the device type.
2 !
2
2 STATUS = READ_HOMEBLOCK (LABEL_STRING[0]);
2
2 MOUNT_OPTIONS[OPT_IS_FILES11] = 1;      ! assume volume is Files-11
2 IF NOT .STATUS
2 AND .STATUS NEQ SS$_INCVOLLABEL
2 THEN ERR_EXIT (.STATUS);
2
3 IF NOT (STATUS = KERNEL_CALL (GET_DEVICE_CONTEXT))
2 THEN ERR_EXIT (.STATUS);
2
2 IF .MOUNT_OPTIONS[OPT_IS_FILES11B]
2 THEN MOUNT_DISK2 ()
2 ELSE MOUNT_DISK1 ();
2
2 ! Rebuild the volume if it was improperly dismounted.
2 !
2
2 IF .CLEANUP_FLAGS[CLF_REBUILD]
```

```
;  398          0927  2       AND .EXE$GL_STATIC_FLAGS [EXE$V_REBLDSYSD]
;  399          0928  2   THEN
;  400          0929  3       BEGIN
;  401          0930  3       ERR_MESSAGE (MOUN$_REBUILD);
;  402          0931  3       STAND_ALONE_REBUILD (.CHANNEL);
;  403          0932  2       END;
;  404          0933  2
;  405          0934  2   IF .LOCK_STATUS [1] NEQ 0
;  406          0935  2   THEN
;  407          0936  3       BEGIN
;  408          0937  3       $DEQ (LKID = .LOCK_STATUS [1]);
;  409          0938  3       LOCK_STATUS [1] = 0;
;  410          0939  2       END;
;  411          0940  2
;  412          0941  2   1
;  413          0942  1   END;                                        ! end of routine MOUNT_COMMAND

                                                                  .TITLE  SYSMOU
                                                                  .IDENT  \V04-000\

                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

        4B  53  49  44  4D  45  54  53  59  53  00000 P.AAB:  .ASCII  \SYSTEMDISK\                        ;
                                                0000A         .BLKB   2
                                     0000000A  0000C P.AAA:   .LONG   10                                  ;
                                     00000000' 00010          .ADDRESS P.AAB                              ;
                          24  55  4F  4D  00014 P.AAC:        .ASCII  \MOU$\                              ;
                                          20  00018           .ASCII  \ \                                 ;
                                          20  00019           .ASCII  \ \                                 ;
                                          20  0001A           .ASCII  \ \                                 ;
                                          20  0001B           .ASCII  \ \                                 ;
                                          20  0001C           .ASCII  \ \                                 ;
                                          20  0001D           .ASCII  \ \                                 ;
                                          20  0001E           .ASCII  \ \                                 ;
                                          20  0001F           .ASCII  \ \                                 ;
                                          20  00020           .ASCII  \ \                                 ;
                                          20  00021           .ASCII  \ \                                 ;
                                          20  00022           .ASCII  \ \                                 ;
                                          20  00023           .ASCII  \ \                                 ;
                                          20  00024           .ASCII  \ \                                 ;
                                          20  00025           .ASCII  \ \                                 ;
                                          20  00026           .ASCII  \ \                                 ;
                                          20  00027           .ASCII  \ \                                 ;
                                          20  00028           .ASCII  \ \                                 ;
                                          20  00029           .ASCII  \ \                                 ;
                                          20  0002A           .ASCII  \ \                                 ;
                                          20  0002B           .ASCII  \ \                                 ;
                                          20  0002C           .ASCII  \ \                                 ;
                                          20  0002D           .ASCII  \ \                                 ;
                                          20  0002E           .ASCII  \ \                                 ;
                                          20  0002F           .ASCII  \ \                                 ;
                                          20  00030           .ASCII  \ \                                 ;
                                          20  00031           .ASCII  \ \                                 ;
                                          20  00032           .ASCII  \ \                                 ;
                                          20  00033           .ASCII  \ \                                 ;
                                        0014  00034 P.AAD:    .WORD   20                                  ;
```

```
               0020  00036              .WORD   32                                          ;
          00000000'  00038              .ADDRESS PHYS_BUFFER                                ;
          00000000'  0003C              .ADDRESS PHYS_NAME                                  ;
               001C  00040              .WORD   28                                          ;
               00EC  00042              .WORD   236                                         ;
           00000000  00044              .LONG   0                                           ;
           00000000  00048              .LONG   0                                           ;
           00000000  0004C              .LONG   0                                           ;

                                        .PSECT  $GLOBAL$,NOEXE,2

                     00000 STORED_CONTEXT::
                                        .BLKB   4
           00000000  00004 MOUNT_FLAGS::
                                        .LONG   0                                           ;
                     00008 LOCK_STATUS::
                                        .BLKB   8
                     00010 DEVICE_INDEX::
                                        .BLKB   4
                     00014 CALLERS_ACMOD::
                                        .BLKB   4
                     00018 CLEANUP_FLAGS::
                                        .BLKB   4
                     0001C CHANNEL::
                                        .BLKB   4
                     00020 MAILBOX_CHANNEL::
                                        .BLKB   4
                     00024 PHYS_BUFFER::
                                        .BLKB   20
           00000000  00038 PHYS_NAME::
                                        .LONG   0                                           ;
          00000000'  0003C              .ADDRESS PHYS_BUFFER                                ;
                     00040 LOG_BUFFER::
                                        .BLKB   20
                     00054 HOME_BLOCK::
                                        .BLKB   512
                     00254 DEVICE_CHAR::
                                        .BLKB   116
                     002C8 DEVICE_CHAR2::
                                        .BLKB   116
                     0033C HOMEBLOCK_LBN::
                                        .BLKB   4
                     00340 HEADER_LBN::
                                        .BLKB   4
                     00344 DEV_INDEX::
                                        .BLKB   4
                     00348 USER_STATUS::
                                        .BLKB   8
                     00350 CURRENT_RVN::
                                        .BLKB   4
                     00354 CURRENT_VCB::
                                        .BLKB   4
                     00358 REAL_RVT::
                                        .BLKB   4
                     0035C REAL_VCB::
                                        .BLKB   4
                     00360 REAL_FCB::
```

```
                                              .BLKB     4
                            00364 REAL_WCB::
                                              .BLKB     4
                            00368 REAL_VCA::
                                              .BLKB     4
                            0036C REAL_AQB::
                                              .BLKB     4
                            00370 LOG_ENTRY::
                                              .BLKB     4
                            00374 MTL_ENTRY::
                                              .BLKB     4
                            00378 SLOG_ENTRY::
                                              .BLKB     4
                            0037C SMTL_ENTRY::
                                              .BLKB     4
                  00000074  00380 DEVCHAR_DESC::
                                              .LONG     116                                              ;
                  00000000' 00384             .ADDRESS  DEVICE_CHAR                                      ;
                  00000074  00388 DEVCHAR_DESC2::
                                              .LONG     116                                              ;
                  00000000' 0038C             .ADDRESS  DEVICE_CHAR2                                     ;
        00424000  C0408300  00390 MOUNT_OPTIONS::
                                              .LONG     -1069513984, 4341760                             ;
                  00000000  00398 PROTECTION::
                                              .LONG     0                                                ;
                  00000000  0039C OWNER_UIC::
                                              .LONG     0                                                ;
                  00000000  003A0 USER_UIC::
                                              .LONG     0                                                ;
                  00000000  003A4 EXTENSION::
                                              .LONG     0                                                ;
                  00000000  003A8 WINDOW::    .LONG     0                                                ;
                  00000000  003AC ACCESSED::
                                              .LONG     0                                                ;
                  00000000  003B0 BLOCKSIZE::
                                              .LONG     0                                                ;
                  00000000  003B4 EXT_CACHE::
                                              .LONG     0                                                ;
                  00000000  003B8 FID_CACHE::
                                              .LONG     0                                                ;
                  00000000  003BC QUO_CACHE::
                                              .LONG     0                                                ;
                  00000000  003C0 EXT_LIMIT::
                                              .LONG     0                                                ;
                  00000001  003C4 DEVICE_COUNT::
                                              .LONG     1                                                ;
                  00000001  003C8 LABEL_COUNT::
                                              .LONG     1                                                ;
                            003CC LOG_NAME::
                                              .BLKB     8
                            003D4 STRUCT_NAME::
                                              .BLKB     8
                            003DC VID_STRING::
                                              .BLKB     8
                            003E4 COMMENT_STRING::
                                              .BLKB     8
                            003EC ACP_STRING::
```

```
                                                          .BLKB    8
                                    003F4 DRIVE_COUNT::
                                                          .BLKB    4
                                    003F8 PARSE_IMP_END::
                                                          .BLKB    0

                                          LABEL_STRING==           P.AAA
                                                          .EXTRN   EXE$GL_STATIC_FLAGS
                                                          .EXTRN   DEV_CTX, EXE$V_REBLDSYSD
                                                          .EXTRN   READ_HOMEBLOCK, MOUNT_DISK1
                                                          .EXTRN   MOUNT_DISK2, STAND_ALONE_REBUILD
                                                          .EXTRN   GET_DEVICE_CONTEXT
                                                          .EXTRN   SYS$GETDVIW, SYS$ENQW
                                                          .EXTRN   SYS$GETCHN, SYS$CMKRNL
                                                          .EXTRN   SYS$DEQ

                                                          .PSECT   $CODE$,NOWRT,2

                                     00FC 00000           .ENTRY   MOUNT_SYSTEM, Save R2,R3,R4,R5,R6,R7      ; 0756
                   57 00000000G   00   9E 00002           MOVAB    LIB$STOP, R7                              ;
                   56     0000'   CF   9E 00009           MOVAB    DEVICE_CHAR, R6                           ;
                   5E       B4    AE   9E 0000E           MOVAB    -76(SP), SP                               ;
  24    AE     0000'   CF   20    28 00012           MOVC3    #32, P.AAC, ALLDEVNAM_BUF                 ; 0794
                            1C    AE   D4 00019           CLRL     ALLDEVNAM_DESC                            ;
            20    AE        24    AE   9E 0001C           MOVAB    ALLDEVNAM_BUF, ALLDEVNAM_DESC+4           ;
        6E     0000'   CF   1C    28 00021           MOVC3    #28, P.AAD, DEVICE_ITMLST                 ; 0815
            10    AE        28    AE   9E 00027           MOVAB    ALLDEVNAM_BUF+4, DEVICE_ITMLST+16         ; 0810
            14    AE        1C    AE   9E 0002C           MOVAB    ALLDEVNAM_DESC, DEVICE_ITMLST+20          ; 0794
                  6D      013D    CF   DE 00031           MOVAL    14$, (FP)                                 ; 0815
          FDC0    C6        02    D0 00036           MOVL     #2, CALLERS_ACMOD                         ; 0840
          FDC8    C6        04    AC   D0 0003B           MOVL     SYS_CHANNEL, CHANNEL                      ; 0841
                            7E    7C 00041           CLRQ     -(SP)                                     ; 0852
                            7E    D4 00043           CLRL     -(SP)                                     ;
                      50    AE    9F 00045           PUSHAB   MOUNT_IOSB                                ;
                      10    AE    9F 00048           PUSHAB   DEVICE_ITMLST                             ;
                            7E    D4 0004B           CLRL     -(SP)                                     ;
                    FDC8    C6    DD 0004D           PUSHL    CHANNEL                                   ;
                            1A    DD 00051           PUSHL    #26                                       ;
          00000000G   00    08    FB 00053           CALLS    #8, SYS$GETDVIW                           ;
                      54    50    D0 0005A           MOVL     R0, STATUS                                ;
                      05    54    E8 0005D           BLBS     STATUS, 1$                                ; 0853
                            54    DD 00060           PUSHL    STATUS                                    ;
                      67    01    FB 00062           CALLS    #1, LIB$STOP                              ;
                1C    AE    04    C0 00065 1$:       ADDL2    #4, ALLDEVNAM_DESC                        ; 0854
                      7E    01    7D 00069           MOVQ     #1, -(SP)                                 ; 0861
                            7E    7C 0006C           CLRQ     -(SP)                                     ;
                            7E    7C 0006E           CLRQ     -(SP)                                     ;
                      34    AE    9F 00070           PUSHAB   ALLDEVNAM_DESC                            ;
                            10    DD 00073           PUSHL    #16                                       ;
                    FDB4    C6    9F 00075           PUSHAB   LOCK_STATUS                               ;
                            05    DD 00079           PUSHL    #5                                        ;
                            1A    DD 0007B           PUSHL    #26                                       ;
          00000000G   00    0B    FB 0007D           CALLS    #11, SYS$ENQW                             ;
                      54    50    D0 00084           MOVL     R0, STATUS                                ;
                      05    54    E8 00087           BLBS     STATUS, 2$                                ; 0862
                            54    DD 0008A           PUSHL    STATUS                                    ;
                      67    01    FB 0008C           CALLS    #1, LIB$STOP                              ;
```

```
                                0134  C6  9F 0008F 2$:   PUSHAB  DEVCHAR_DESC2                                    ; 0871
                                      7E  D4 00093        CLRL    -(SP)                                            ;
                                012C  C6  9F 00095        PUSHAB  DEVCHAR_DESC                                     ;
                                      7E  D4 00099        CLRL    -(SP)                                            ;
                                FDC8  C6  DD 0009B        PUSHL   CHANNEL                                          ;
              00000000G   00          05  FB 0009F        CALLS   #5, SYS$GETCHN                                   ;
  74   A6                 66    0074  8F  29 000A6        CMPC3   #116, DEVICE_CHAR, DEVICE_CHAR2                  ; 0873
                                      05  12 000AD        BNEQ    3$                                               ;
  08          01          A6          06  E0 000AF        BBS     #6, DEVICE_CHAR+1, 4$                            ; 0874
                          7E    01CC  8F  3C 000B4 3$:   MOVZWL  #460, -(SP)                                      ; 0875
                          67          01  FB 000B9        CALLS   #1, LIB$STOP                                     ;
  07          02          A6          02  E0 000BC 4$:   BBS     #2, DEVICE_CHAR+2, 5$                            ; 0877
                          7E      84  8F  9A 000C1        MOVZBL  #132, -(SP)                                      ; 0878
                          67          01  FB 000C5        CALLS   #1, LIB$STOP                                     ;
  07          02          A6          03  E1 000C8 5$:   BBC     #3, DEVICE_CHAR+2, 6$                            ; 0880
                          7E      6C  8F  9A 000CD        MOVZBL  #108, -(SP)                                      ; 0881
                          67          01  FB 000D1        CALLS   #1, LIB$STOP                                     ;
  08                      66          05  E1 000D4 6$:   BBC     #5, DEVICE_CHAR, 7$                              ; 0883
                          7E    01CC  8F  3C 000D8        MOVZWL  #460, -(SP)                                      ; 0884
                          67          01  FB 000DD        CALLS   #1, LIB$STOP                                     ;
                               0000'  CF  9F 000E0 7$:   PUSHAB  LABEL_STRING                                     ; 0909
                  0000G   CF          01  FB 000E4        CALLS   #1, READ_HOMEBLOCK                               ;
                          54          50  D0 000E9        MOVL    R0, STATUS                                       ;
                   0140   C6          02  88 000EC        BISB2   #2, MOUNT_OPTIONS+4                              ; 0911
                          0E          54  E8 000F1        BLBS    STATUS, 8$                                       ; 0912
              0000010C    8F          54  D1 000F4        CMPL    STATUS, #268                                     ; 0913
                                      05  13 000FB        BEQL    8$                                               ;
                                      54  DD 000FD        PUSHL   STATUS                                           ; 0914
                          67          01  FB 000FF        CALLS   #1, LIB$STOP                                     ;
                                      7E  D4 00102 8$:   CLRL    -(SP)                                            ; 0916
                                      5E  DD 00104        PUSHL   SP                                               ;
                               0000G  CF  9F 00106        PUSHAB  GET_DEVICE_CONTEXT                               ;
              00000000G   9F          03  FB 0010A        CALLS   #3, @#SYS$CMKRNL                                 ;
                          54          50  D0 00111        MOVL    R0, STATUS                                       ;
                          05          54  E8 00114        BLBS    STATUS, 9$                                       ;
                                      54  DD 00117        PUSHL   STATUS                                           ; 0917
                          67          01  FB 00119        CALLS   #1, LIB$STOP                                     ;
  07           0140       C6          02  E1 0011C 9$:   BBC     #2, MOUNT_OPTIONS+4, 10$                         ; 0919
               0000G      CF          00  FB 00122        CALLS   #0, MOUNT_DISK2                                  ; 0920
                                      05  11 00127        BRB     11$                                              ;
               0000G      CF          00  FB 00129 10$:  CALLS   #0, MOUNT_DISK1                                  ; 0921
  22           FDC5       C6          01  E1 0012E 11$:  BBC     #1, CLEANUP_FLAGS+1, 12$                         ; 0926
  16 00000000G            00 00000000G 8F E1 00134        BBC     #EXE$V_REBLDSYSD, EXE$GL_STATIC_FLAGS, 12$       ; 0927
                            0072A01B  8F  DD 00140        PUSHL   #7512091                                         ; 0930
     00000000G            00          01  FB 00146        CALLS   #1, LIB$SIGNAL                                   ;
                                FDC8  C6  DD 0014D        PUSHL   CHANNEL                                          ; 0931
               0000G      CF          01  FB 00151        CALLS   #1, STAND_ALONE_REBUILD                          ;
                          50    FDB8  C6  D0 00156 12$:  MOVL    LOCK_STATUS+4, R0                                ; 0934
                                      11  13 0015B        BEQL    13$                                              ;
                                      7E  7C 0015D        CLRQ    -(SP)                                            ; 0937
                                      7E  D4 0015F        CLRL    -(SP)                                            ;
                                      50  DD 00161        PUSHL   R0                                               ;
     00000000G            00          04  FB 00163        CALLS   #4, SYS$DEQ                                      ;
                                FDB8  C6  D4 0016A        CLRL    LOCK_STATUS+4                                    ; 0938
                          50          01  D0 0016E 13$:  MOVL    #1, R0                                           ; 0942
                                          04 00171        RET                                                      ;
                                        0000 00172 14$:  .WORD   Save nothing                                     ; 0815
```

```
SYSMOU                                                H 5
V04-000                                     16-Sep-1984 02:12:43    VAX-11 Bliss-32 V4.0-742          Page  14
                                            14-Sep-1984 13:16:57    [SYSINI.SRC]SYSMOU.B32;1                (3)

                                   7E  D4 00174          CLRL    -(SP)                                     ;
                                   5E  DD 00176          PUSHL   SP                                        ;
                  7E       04      AC  7D 00178          MOVQ    4(AP), -(SP)                              ;
          0000V   CF               03  FB 0017C          CALLS   #3, MAIN_HANDLER                          ;
                                       04 00181          RET                                               ;

; Routine Size:  386 bytes,    Routine Base:  $CODE$ + 0000
```

```
_$2

Sym
---
EVT
EXE
EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE
FAI
FAI
FOR
GAI
INI
INI
IOC
IOC
IOC
IOC
JOI
LCK
LCK
LCK
LCK
LCK
LCK
LCK
```

```
1 ROUTINE MAIN_HANDLER (SIGNAL, MECHANISM) =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine is the main level condition handler for the MOUNT
1 !       utility. It undoes anything that MOUNT has done so far and then
1 !       unwinds and returns the condition code as status to MOUNT's
1 !       caller (i.e., the CLI).
1 !
1 !
1 ! CALLING SEQUENCE:
1 !       MAIN_HANDLER (ARG1, ARG2)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of signal array
1 !       ARG2: address of mechanism array
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       SS$_CONTINUE
1 !
1 ! SIDE EFFECTS:
1 !       stack unwound, control passed to CLI
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       SIGNAL          : REF BBLOCK,   ! signal array
2       MECHANISM       : REF BBLOCK;   ! mechanism array
2 EXTERNAL
2       USER_STATUS     : VECTOR;       ! status return of some routines
2
2
2 ! Do cleanup as indicated by the status flags.
2 ! Cause the condition code to be returned in R0 as the main routine value.
2 !
2
2 IF .SIGNAL[CHF$L_SIG_NAME] NEQ SS$_UNWIND
2 AND .BBLOCK [SIGNAL[CHF$L_SIG_NAME], STS$V_SEVERITY] EQL STS$K_SEVERE
2 THEN
3     BEGIN
3
3     IF .SIGNAL[CHF$L_SIG_NAME] NEQ 0
3     THEN MECHANISM[CHF$L_MCH_SAVR0] = .SIGNAL[CHF$L_SIG_NAME]
```

```
;  472        1000  3         ELSE MECHANISM[CHF$L_MCH_SAVR0] = .USER_STATUS[0];
;  473        1001  3         IF .LOCK_STATUS [1] NEQ 0
;  474        1002  3         THEN
;  475        1003  4             BEGIN
;  476        1004  4             $DEQ (LKID = .LOCK_STATUS [1]);
;  477        1005  4             LOCK_STATUS [1] = 0;
;  478        1006  3             END;
;  479        1007  3         $UNWIND ();
;  480        1008  2         END;
;  481        1009  2
;  482        1010  2     RETURN SS$_CONTINUE;                          ! continue from success signals
;  483        1011  2
;  484        1012  1     END;                                          ! end of routine MAIN_HANDLER


                                                                            .EXTRN  SYS$UNWIND

                                                       0000 00000 MAIN_HANDLER:
                                                                            .WORD   Save nothing                          ; 0943
                                        51          04   AC   D0 00002      MOVL    SIGNAL, R1                            ; 0993
                           00000920     8F          04   A1   D1 00006      CMPL    4(R1), #2336                          ;
                                                         3F   13 0000E      BEQL    4$                                    ;
       04          04   A1              03               00   ED 00010      CMPZV   #0, #3, 4(R1), #4                     ; 0994
                                                         37   12 00016      BNEQ    4$                                    ;
                                        50          08   AC   D0 00018      MOVL    MECHANISM, R0                         ; 0999
                                                    04   A1   D5 0001C      TSTL    4(R1)                                 ; 0998
                                                         07   13 0001F      BEQL    1$                                    ;
                                  0C    A0          04   A1   D0 00021      MOVL    4(R1), 12(R0)                         ; 0999
                                                         06   11 00026      BRB     2$                                    ;
                                  0C    A0       0000G   CF   D0 00028 1$:  MOVL    USER_STATUS, 12(R0)                   ; 1000
                                        50       0000'   CF   D0 0002E 2$:  MOVL    LOCK_STATUS+4, R0                     ; 1001
                                                         11   13 00033      BEQL    3$                                    ;
                                                         7E   7C 00035      CLRQ    -(SP)                                 ; 1004
                                                         7E   D4 00037      CLRL    -(SP)                                 ;
                                                         50   DD 00039      PUSHL   R0                                    ;
                           00000000G    00               04   FB 0003B      CALLS   #4, SYS$DEQ                           ;
                                                 0000'   CF   D4 00042      CLRL    LOCK_STATUS+4                         ; 1005
                                                         7E   7C 00046 3$:  CLRQ    -(SP)                                 ; 1007
                           00000000G    00               02   FB 00048      CALLS   #2, SYS$UNWIND                        ;
                                        50               01   D0 0004F 4$:  MOVL    #1, R0                                ; 1010
                                                              04 00052      RET                                           ; 1012

; Routine Size:  83 bytes,    Routine Base:  $CODE$ + 0182


;  485        1013  1
;  486        1014  1 END
;  487        1015  0 ELUDOM



                                                                            .EXTRN  LIB$SIGNAL, LIB$STOP

;                                 PSECT SUMMARY
;
```

```
_$2

Sym
---
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LCK
LKI
LKI
LKI
LKI
LKI
LKI
LKI
LKI
LKI
LKI
LOS
MEM
MMG
OPA
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PMS
PR1
```

```
                                                                    K  5
SYSMOU                                                          16-Sep-1984 02:12:43     VAX-11 Bliss-32 V4.0-742                Page  17
V04-000                                                         14-Sep-1984 13:16:57     [SYSINI.SRC]SYSMOU.B32;1                      (4)

;       Name                        Bytes                       Attributes
;
;  $GLOBAL$                            1016  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                                80  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                               469  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




;                                 Library Statistics
;
;                                                 -------- Symbols --------        Pages         Processing
;       File                                      Total    Loaded   Percent       Mapped        Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1                  18619       37         0        1000           00:01.9




;                                          COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:SYSMOU/OBJ=OBJ$:SYSMOU MSRC$:SYSMOU/UPDATE=(ENH$:SYSMOU)

; Size:          469 code + 1096 data bytes
; Run Time:         00:18.5
; Elapsed Time:     00:36.7
; Lines/CPU Min:      3293
; Lexemes/CPU-Min: 30395
; Memory Used:  169 pages
; Compilation Complete
```

```
_$2

Sym
---

PR$
QDC
QDD
QDF
QDI
QDR
QDT
QDW
QDW
QUC
QUC
RCV
REA
REA
REC
REC
REQ
SCH
SCH
SCH

SCH
SCH
SCH
SCH
SCH
SCH
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SCS
SEN
SND
SS$
SS$
SS$
SS$
SS$
```

0390 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY